

---

## ✓ Practice Interview

### Objective

*The partner assignment aims to provide participants with the opportunity to practice coding in an interview context. You will analyze your partner's Assignment 1. Moreover, code reviews are common practice in a software development team. This assignment should give you a taste of the code review process.*

### Group Size

Each group should have 2 people. You will be assigned a partner

### Part 1:

You and your partner must share each other's Assignment 1 submission.

## ✓ Part 2:

Create a Jupyter Notebook, create 6 of the following headings, and complete the following for your partner's assignment 1:

---

- Paraphrase the problem in your own words.
  - Find all the missing numbers in a list which contains integers ranging from 0 to n (maximum value).
  - Identify and return a list of all numbers within this range that are not present in the given list.
  - If every number in the range  $[0, n]$  is present in the list, you should return -1.
- 

- Create 1 new example that demonstrates you understand the problem. Trace/walkthrough 1 example that your partner made and explain it.

My Example :

- lst = [0, 1, 3, 4]
- The Output is [2]
- The range is [1 , 4] and the missing number is 2.

Daya's Example :

- lst = [1, 2, 3, 4, 5]
- The Output is [0]
- The range is [0 , 5]. The missing number is 0

- Copy the solution your partner wrote.

```
def missing_numbers(lst):
    n = len(lst)
    ideal_set = set(range(n + 1))
    lst_set = set(lst)
    missing = sorted(list(ideal_set - lst_set))

    if not missing:
        return -1

    return missing
```

- Explain why their solution works in your own words.
- The solution uses Python's set properties to help identify the missing numbers .
- The expected numbers are created in 'ideal\_set' as a range from '0' to the maximum value 'n'.
- The input list 'lst' converted to a set to remove duplicates and make searching faster.
- The difference between the expected set and the input set yields the missing numbers.
- sorted function sort the output in ascending order
- If there are no missing numbers '-1' is returned.

- Explain the problem's time and space complexity in your own words.

Time complexity:

- $O(n)$  for generating the sets and calculating the difference between them.
- $O(m \log m)$  for sorting the list of missing numbers, where  $m$  is the missing numbers that are less than or equal to  $n$ .

Space Complexity : The space complexity is space required for each set which is  $O(n)$ .

The space complexity of  $O(n)$  and the sorting step's  $O(n \log n)$  time complexity limit the efficiency of the solution for larger datasets.

---

- Critique your partner's solution, including explanation, and if there is anything that should be adjusted.

Daya did not explain the time complexity for sorting the list of missing numbers, which is  $O(n \log n)$ , and how it affects the efficiency of the solution for large datasets.

---

## ✓ Part 3:

---

Please write a 200 word reflection documenting your process from assignment 1, and your presentation and review experience with your partner at the bottom of the Jupyter Notebook under a new heading "Reflection." Again, export this Notebook as pdf.

## ✓ Reflection

Daya's solution for finding missing numbers in a list of integers from 0 to  $n$  is a clear and efficient approach. By taking advantage of using set properties, the initial solution computes the difference between the ideal range and the input list, and identifying missing numbers in a direct way. However, the space complexity of  $O(n)$  and the sorting step's  $O(n \log n)$  time complexity limit its efficiency for larger datasets.

Daya proposes using an alternative approach to address these limitations. This approach takes an advantage of in-place marking. By negating values at indices corresponding to the numbers in the list, this approach reduces space complexity to  $O(1)$  and achieves a linear time complexity of  $O(n)$ . This makes it suitable for large datasets.

Another alternative solution that uses boolean array can also solve the problem in  $O(n)$  time, but with  $O(n)$  space complexity. While less space-efficient than in-place marking, it offers a more intuitive approach for some users.

In conclusion, Daya's solution highlights the importance of considering both time and space complexity when choosing an algorithm. The in-place marking solution is generally preferred for large datasets due to its optimal space complexity, but the boolean array approach can be a simpler alternative in some cases. Understanding these trade-offs allows for selecting the most appropriate solution for a given problem and dataset size.



---

## Evaluation Criteria

We are looking for the similar points as Assignment 1

- Problem is accurately stated
- New example is correct and easily understandable
- Correctness, time, and space complexity of the coding solution
- Clarity in explaining why the solution works, its time and space complexity
- Quality of critique of your partner's assignment, if necessary

## Submission Information

 **Please review our [Assignment Submission Guide](#)**  for detailed instructions on how to format, branch, and submit your work. Following these guidelines is crucial for your submissions to be evaluated correctly.

### Submission Parameters:

- Submission Due Date: HH:MM AM/PM - DD/MM/YYYY
- The branch name for your repo should be: assignment-2
- What to submit for this assignment:
  - This Jupyter Notebook (assignment\_2.ipynb) should be populated and should be the only change in your pull request.
- What the pull request link should look like for this assignment: <https://github.com/your-github-username/algorithms-and-data-structures/pull/your-id>

<your\_github\_username>/algorithms\_and\_data\_structures/pull/<pr\_id>

- Open a private window in your browser. Copy and paste the link to your pull request into the address bar. Make sure you can see your pull request properly. This helps the technical facilitator and learning support staff review your submission easily.

#### Checklist:

- ☐ Created a branch with the correct naming convention.
- ☐ Ensured that the repository is public.
- ☐ Reviewed the PR description guidelines and adhered to them.
- ☐ Verify that the link is accessible in a private browser window.

If you encounter any difficulties or have questions, please don't hesitate to reach out to our team via our Slack at `#cohort-3-help`. Our Technical Facilitators and Learning Support staff are here to help you navigate any challenges.